The Journal of
COMPUTERS IN MATHEMATICS AND SCIENCE TEACHING

Spring 1983 **Volume II, Number 3**

FEATURES

DEPARTMENTS

On the Cover: A graphic display of phases in the simulated flight of the Space Shuttle from the program RENDEZVOUS by Edu-Ware, P.O. Box 22222, Agoura, CA 91301. See related science simulation article, "Science-Based Simulation Development: An Example in Physics." Staff photo.

The Journal of Computers in Mathematics and Science Teaching (ISSN 0731-9258) is published quarterly in the Fall, Winter, Spring and Summer by the Association for Computers in Mathematics and Science Teaching, a non-profit educational organization. ACMST membership is $15 a year and includes a subscription to *The Journal of Computers in Mathematics and Science Teaching*. Single copy price is $4.00. Manuscripts submitted for publication consideration and request for permission to reprint should be sent to Gary H. Marks, Editor. *JCMST* Author's Guidelines are available on request. All correspondence should be addressed to *The Journal of Computers in Mathematics and Science Teaching*, P.O. Box 4455, Austin, TX 78765. Articles are the views of the authors and do not necessarily represent Association policy.

Interfacing the Atari Microcomputer in the Science Laboratory

Michael McInerney and Kenneth Williams

The computer has proven to be a useful aid in the teaching of science (Tinker, 1978; Eisenkraft, 1981). In the laboratory students may now simulate experiments before tackling them and compare theoretical and practical results after the experiment. Microcomputers may also be used to control experiments and to gather data automatically (Rafert and Nicklin, 1982).

The use of microcomputers to control experiments and gather data is confined, at present, mainly to advanced science laboratories. Microcomputers are seldom used in introductory laboratories because the electronics involved are thought to be too difficult. Indeed, the setting up of a popular single board computer like the KIM to log data is a formidable project for someone not thoroughly acquainted with the machine. In the last few years, however, the manufacturers of many home computers have made their machines exceedingly simple to interface. They have not, of course, incorporated this ability so that science teachers can use their machines in the laboratory; but rather for computer games.

The game ports of microcomputers are quite simple to interface. In this paper we describe how this interfacing is performed on the Atari 800, though the techniques described are easily transferred to other microcomputers.

The Atari Game Port

The Atari game port has nine pins. Figure 1 shows the pin arrangement as you would see it looking at the port from the outside of the computer.

We found it difficult to access the game port pins directly from the computer console because the pins are so close together. In particular the +5V and ground pins are next to each other and a short circuit is only too easy to accomplish. To solve this access difficulty we built a simple extension board. This wooden board has nine sockets arranged in the same pattern as the pins in the game port, and each socket is connected to the corresponding game port pin via a lead from the board.

The game port allows both digital and analog input and digital output. The paddle pins are the analog inputs, the joystick and paddle trigger pins are the digital input while the joystick pins allow digital output.

Digital Input

The digital game port pins are latched high when the computer is switched on and in this state return a value of 1. The value returned from the pins changes to a 0 when the

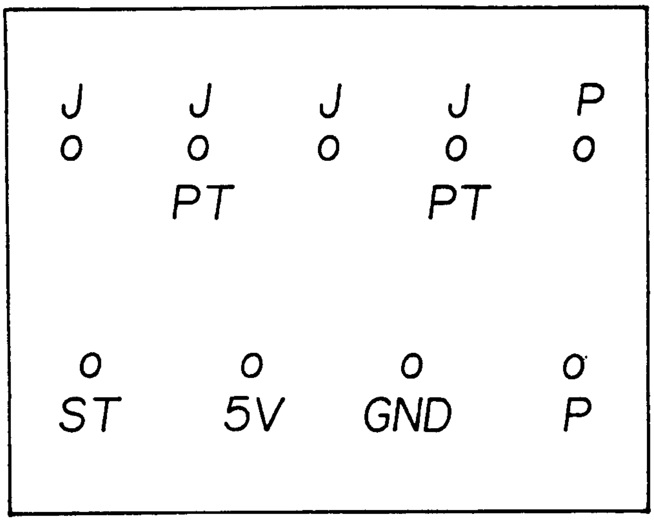

Figure 1. Game Port configuration of the Atari 800 and 400

Michael McInerney is an assistant professor, Physics Department, Eastern Illinois University, Charleston, IL 61920. Kenneth Williams is an engineer, Applied Physics Laboratory, Johns Hopkins University, Baltimore, MD 21239.

TABLE 1

Pin Name	*Connection*	*Basic Access Word*
J: Joystick	J/GND	STICK(N)
PT: Paddle trigger	PT/GND	PTRIG 2N or PTRIG(2N +1)
P: Paddle	P/+5V	PADDLE(2N) or PADDLE(2N +1)
ST: Joystick trigger	ST/GND	STRIG(N)

Note that N is the number between 0 and 3 of the game port.

pins are brought low by connection to GND. This facility is easily demonstrated by connecting a jumper wire across STRIG and GND of game port 0, and running the following program in BASIC.

```
10 PRINT STRIG (0)
20 GOTO 10
```

The digital pins divide naturally into two groups: the joystick pins and the STRIG trigger pins.

The joystick pins in each game port may all be accessed by one BASIC command: READ STICK(N). N is a number from 0 to 3 depending on the game port being used. The number returned by this command is the decimal equivalent of the binary number derived by reading the four joystick pins with each joystick pin forming a bit of a four bit binary number. Within each of these four bits, J0 and J1 of game port 0 are connected to GND the command READ STICK(0) would return the number 12, binary 1100.

Two of the joystick pins, J3 and J2, in each of the game ports may be accessed individually by the commands READ PTRIG(2N) and READ PTRIG(2N + 1), respectively. Once again N is the number of the game port.

The joystick trigger pins, STRIG on Diagram 1, are read from BASIC by the command READ STRIG(N); N is the game port number. In this sense they are the same as the PTRIG pins; however, there is a potentially useful difference in that STRIG may be latched (see Atari manual, 1980). If bit 2 of the word in memory address (53277) is set to 1, STRIG is brought low (connected to GND) the STRIG reading remains 0 even if STRIG goes high again. This facility allows the computer to note, at its leisure, that an event has occurred.

Digital Output

The four game ports of the Atari contain a total of 16 joystick pins. Within the computer these pins connect to the two ports of a programmable input-output chip (PIA) as shown in Diagram 2.

In essence the PIA acts as a buffer between the data bus of the computer and the outside world (Leventhal, 1979). Although the PIA has two ports only one of these ports may be connected to the data bus at any one time. Each port has its own address, PORTA = 54016 and PORTB = 54017 respectively. Data enters or leaves each port through the data register associated with that port.

As its name implies, the PIA is a programmable chip. Each port line may be programmed as an output or an input. To hold this program information each port has an eight bit direction register, each bit of which corresponds to a line of the port. If a bit in the direction register is set to one, the corresponding line is an output; if the bit is set to a zero, the corresponding line is an input.

In order to be used for general input-output operations the PIA is equipped with two eight bit control registers, one for each port. These registers, called PACTL and PBCTL with addresses (54018) and (54019) control the A and B ports, respectively. The control lines of the PIA chip are not accessible outside of the ATARI computer; therefore, most of the contents of the control registers are uninteresting to us. For reasons perhaps to do with the number of pins available on the standard chip, the data direction register and data register of each port have the same address. In order to distinguish between each of these registers bit 2 of the control register must be set to select the data register, or cleared to select the direction register.

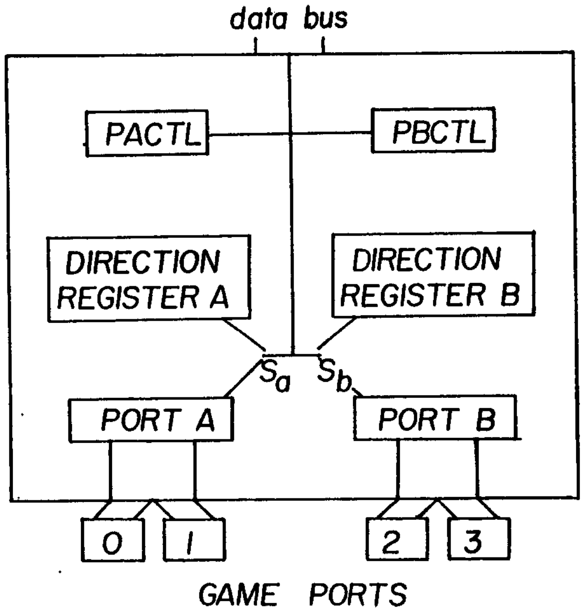

Figure 2. Schematic of the PIA chip. S_a and S_b are switches controlled, respectively, by PACTL and PBCTL. Solid lines represent data paths.

The following steps are required to output through PIA port A:

a. Bit 2 of PACTL must be set to 1 so that the address PORTA reads into the direction register of the PIA.
b. A number is read into the direction register. This number is the decimal equivalent of the 8 bit binary number formed by placing a 1 to correspond to a line out and a 0 to correspond to a line in.
c. Bit 2 of PACTL is set to 0 so that PORTA addresses the data register.
d. The data to be output is moved to PORTA.

An example of this method is the following program which configures PORTA as an output port and outputs '0' on all lines. On start-up the computer latches all lines high, corresponding to a 1. Thus on running this program you should see all lines go from high to low, from 1 to 0. The lines will remain low until a new command is sent to the data register.

```
10 PORTA = 54016 : PACTL = 54018
20 POKE PACTL, 48: REM DIRECTION
   REGISTER SELECTED AT PORTA
30 POKE PORTA, 255 : REM ALL PINS SET AS
   OUTPUT
```

40 POKE PACTL, 52 : REM CHANGE TO OUTPUT REGISTER AT PORTA
50 POKE PORTA, 255 : REM OUTPUT 1 ON ALL LINES

Note that although many numbers will set and reset bit 2 of PACTL we are using 48 and 52 respectively. The choice of the numbers 48 and 52 is directed by the particular functioning of the Atari microcomputer.

When the 'Reset' button on the computer is pressed, the registers of the PIA are cleared configuring the ports as input ports and connecting addresses PORTA and PORTB with the corresponding data registers.

There is one drawback with the digital output facility just described; the Atari will not allow enough current to be drawn through the joystick pins to light even an LED. We have overcome this problem (McInerney and Williams) by the use of a simple amplifier. In essence each output line is connected, via buffers, to the base pin of a Darlington power transistor. If suitably biased, current will flow through the Darlington when the output of the line from the data register is high (McInerney and Williams).

Analog Input

Another method of interfacing through the Atari game ports is the paddle input, so named because of its use as a control for Pong and other games involving the movement of an electronic paddle on the television monitor. the paddle controller used in such games is simply a linear potentiometer, so that the paddle inputs are, in effect, resistance-reading interfaces. They are, perhaps the most useful of the game-port interfaces, in terms of the amount of information that can be accepted by the microcomputer (See Tinker, 1982).

How the Paddle Function Works

The Atari paddle input is designed to be used with a 1MΩ linear potentiometer. One end of the potentiometer is connected to the Atari's +5V power supply via pin 7 of the game port. The tap is connected to one of the paddle inputs (pins 5 and 9).

The microcomputer 'reads' the potentiometer resistance by utilizing the R-C arrangement shown in Figure 3. The capacitance C, located inside the microcomputer, is fixed in value, so that the rise time R X C of the voltage across the capacitance is directly proportional to the external potentiometer resistance R. The timer, shown in the circuit as a part of the POKEY chip, monitors the rise time of the capacitance voltage and generates a squarewave timing pulse. The timing signal is initiated by a zero-voltage level across the capacitance and cut off when the voltage reaches a threshold level, normally about ⅔ of the +5V available. The timer then dumps the charge on the capacitance so that the voltage drops back to zero and another reading can be taken.

The timing pulses are relayed to the microprocessor via the timing line. The length of the timing pulse, directly related to the rise time and, therefore, resistance read, controls a counter which increments itself each time a line is scanned on the television monitor. The counter functions over the duration of the pulse, dumps the count into a location in RAM and resets to zero for the next pulse.

The RAM locations containing counts are accessible through the Atari-BASIC PADDLE functions. There are eight such functions, one for each of the two paddle inputs on the four game ports of the Atari 800. The PADDLE-function values range from 0 to 228, as the potentiometer resistance ranges from 0 to 1MΩ, correspondingly. The maximum value of 228 corresponds to the 228 scan lines available per television frame (See Elberfeld, 1982).

Transducers

While the Atari paddle inputs are designed to be used with a particular potentiometer, it also is possible to use them with a variety of devices which measure some analog property, such as light intensity, position or temperature, or other electrical properties. Such devices, called transducers, include photocells, light-sensitive transistors, thermistors and piezoelectric devices, as well as potentiometers. It is possible to interface these devices with the Atari through the paddle inputs in the same fashion as the potentiometer, although some modification is needed in certain cases. In all instances, the transducer must have a large resistance or resemble one.

A few examples of appropriate transducers are the photocell, the thermistor, and the phototransistor.

Photocell. A highly sensitive cadmium sulfide photocell can be substituted for the potentiometer in Figure 3 to serve as a light detector for spectroscopic measurements. A PADDLE-function value of 0 corresponds to no resistance in the photocell, which occurs when the light intensity is high. On the other hand, a value of 228 indicates high resistance, and therefore low light intensity. The maximum resistance of the photocell should, of course, be around 1MΩ to be compatible with the function of the Atari game ports. Also, although it is highly sensitive, the photocell responds too slowly to intensity changes to be of use in photogate timing experiments.

Thermistor. Thermistors work well as a temperature sensor when connected between a Paddle pin and the +5V of the gameport. The sensitivity of the thermistor over a few degrees temperature range is rather small but may easily be enhanced by the methods of section 4.6.

Phototransistor. A phototransistor is generally less stable than a photocell, but has a rapid response time that is appropriate for photogate timing of moving bodies. The FPT 100 silicon phototransistor has an emitter-collector current

which varies directly with the intensity of light falling on it. Because of its inherent instability, it may be desirable to modify the circuit, as indicated in Figure 3. To reduce the uncertainty in the reading to a few percent, a 3-ΩP capacitor is placed in parallel with a fixed resistance between +5V and the paddle input. The phototransistor is then placed between the paddle input and ground (pin 8). In this configuration a high light intensity corresponds to a PADDLE-function value of 228, in contrast to the photocell arrangement.

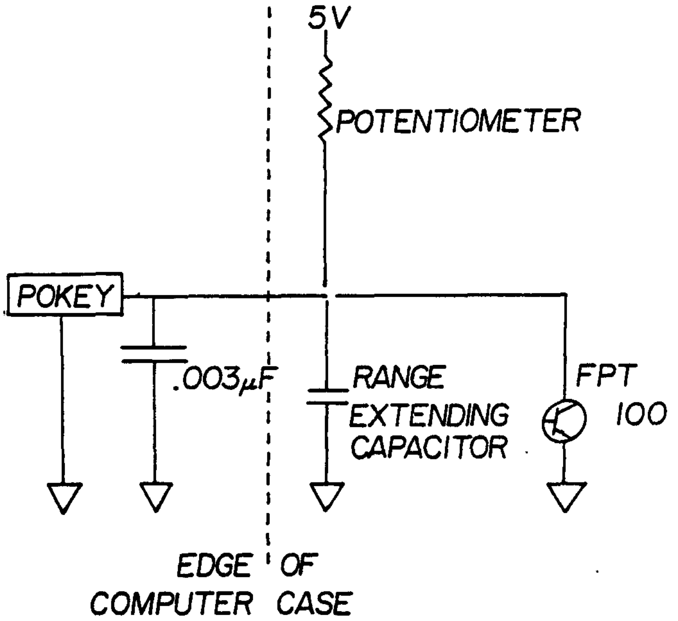

Figure 3. Paddle port schematic.

Problems of Resolution and Timing

There are several problems associated with the paddle inputs. For example, depending on the resistance read, it can take up to 16 msec for the PADDLE function to respond to any change in the transducer resistance. Also, with only 228 values possible for PADDLE, it is often difficult to achieve the desired resolution of data. If the resistance of the transducer is not in the range from 0 to 1MΩ, the number of possible values may be decreased or the PADDLE function may not respond at all over a significant portion of the desired range of operation. All of these problems can be alleviated somewhat through modifications, either in the external circuit or by software.

The problem of slow response results from the time required for the capacitor to charge to its threshold voltage. It is possible, however, to place the microcomputer in a fast-scan mode, in which the threshold voltage is lowered so that the timing pulses from POKEY are shortened considerably. Also, the scan line counter works much more rapidly. Although some accuracy is lost by lowering the threshold voltage, a full count of 228 now corresponds to the time required to scan only two lines, instead of a full frame.

The fast-scan mode is initiated by turning on bit 2 in the SKCTL control register, located at decimal address 53775 in RAM. POKEY will not automatically dump charge from the capacitor in this mode. It is necessary to reset SKCTL to the normal mode after the reading is taken. In order to be effective, fast-scan must be done in machine language.

The poor resolution of the PADDLE function can be a serious drawback. If a potentiometer is used to determine angular position, the accuracy of the measurement is not even to the nearest degree. One way to improve the resolution in such a case is to use a combination of PADDLE functions. For instance, one can connect the potentiometer tap to the +5V and each end to a different paddle input. By subtracting one PADDLE function from the other, it is now possible to obtain values in the range of −228 to +228, thus doubling the resolution. Odd-numbered values result from the two PADDLE functions being out of phase as a result of the finite width of the potentiometer tap.

Finally, the problem of range can be solved most conveniently by remembering that the Atari paddle input operates on the principle of a rise time associated with an R-C circuit. If the resistance range is too small, then the effective rise times can be maintained by making the capacitance correspondingly larger. This can be done by connecting an additional capacitor between ground and the paddle input. The positioning of this capacitor is shown in Figure 3.

Conclusion

We have shown in this paper how simple it is to input and output through the game ports of the Atari microcomputer. The only real drawback of gameport interfacing as compared to the more usual laboratory interfacing is that there is no way to initiate an interrupt of the microprocessor through the gameports. In many laboratory applications where the computer is dedicated to one particular experiment, this lack of an interrupt is not particularly serious. However, where interrupts are essential a connection through the RS232 port on the side of the computer may be used.

Gameport interfacing makes possible the use of microcomputers to gather data even in introductory science laboratories.

Bibliography

1. *Atari Personal Computer System Hardware Manual,* Atari, Inc., 1980, p. II. 31.
2. Eisenkraft, A. "Microcomputers in a Physics Curriculum," *The Journal of Computers in Mathematics and Science Teaching* 1, No. 2, p. 9, Winter, 1981.
3. Elberfeld, J. "Apple Game Paddles," *Compute,* July, 1982, vol. 4, no. 7, p. 92–94.
4. Leventhal, Lance A. *6502 Assembly Language Programming,* Osborne/McGraw-Hill, Berkeley, California, 1979.

5. McInerney, M. and Williams, K. E. "Game Port Output for the Atari 800-400," unpublished.
6. Rafert, Bruce and Nicklin, R. C. "Microcomputers in the Laboratory," *American Journal of Physics 50,* 108 February, 1982.
7. Tinker, R. "Low Cost Science Experiments using Micros," *Hands On,* Spring, 1982, p. 8.
8. Tinker, Robert F. and Stringer, Gene A. "Microcomputers: Applications to Physics Teaching," *The Physics Teacher, 16,* 1978, p. 436.